

BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTTTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL

```
BBBBBBBBB      AAAAAA      SSSSSSSSS  EEEEEEEEEEE  NN      NN  DDDDDDDDD
BBBBBBBBB      AAAAAA      SSSSSSSSS  EEEEEEEEEEE  NN      NN  DDDDDDDDD
BB      BB  AA      AA  SS      SS      EE      NN      NN  DD      DD
BB      BB  AA      AA  SS      SS      EE      NN      NN  DD      DD
BB      BB  AA      AA  SS      SS      EE      NNNN     NN  DD      DD
BBBBBBBBB      AA      AA  SSSSSS      EEEEEEEE      NN  NN  NN  DD      DD
BBBBBBBBB      AA      AA  SSSSSS      EEEEEEEE      NN  NN  NN  DD      DD
BB      BB  AAAAAAAAAA      SS      EE      NN      NNNN     DD      DD
BB      BB  AAAAAAAAAA      SS      EE      NN      NNNN     DD      DD
BB      BB  AA      AA  SS      EE      NN      NN      NN  DD      DD
BB      BB  AA      AA  SS      EE      NN      NN      NN  DD      DD
BBBBBBBBB      AA      AA  SSSSSSSSS  EEEEEEEEEEE  NN      NN  DDDDDDDDD
BBBBBBBBB      AA      AA  SSSSSSSSS  EEEEEEEEEEE  NN      NN  DDDDDDDDD
                                     ....
                                     ....
                                     ....
                                     ....
```

```
LL      IIIIII      SSSSSSSSS
LL      IIIIII      SSSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLL  IIIIII      SSSSSSSSS
LLLLLLLLLLL  IIIIII      SSSSSSSSS
```



```

1 0001 0 MODULE BASSEND (
2 0002 0 IDENT = '1-018'
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY: BASIC-PLUS-2 Frame Support
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 These routines set up and tear down frames for BASIC-PLUS-2.
37 0037 1 Frames are used for main routines, external functions,
38 0038 1 external subroutines, internal functions (both DEFs and DEF*s)
39 0039 1 internal subroutines (GOSUBs) and condition handlers.
40 0040 1
41 0041 1 ENVIRONMENT: VAX-11 user mode
42 0042 1
43 0043 1 AUTHOR: John Sauter, CREATION DATE: 10-Oct-78
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1 1-001 - Original.
48 0048 1 1-002 - Change from BASS$ to BSF$ for stack frame names. JBS 08-FEB-1979
49 0049 1 1-003 - Purge any FIELD variables being deallocated. JBS 28-FEB-1979
50 0050 1 1-004 - Deallocate dynamic strings, but ignore fixed string templates.
51 0051 1 JBS 20-MAR-1979
52 0052 1 1-005 - Remove edit 003--the call to purge FIELD variables will be
53 0053 1 issued by the compiler so that the FIELD code need not be
54 0054 1 in the sharable library. JBS 02-APR-1979
55 0055 1 1-006 - Similarly, don't flush virtual arrays. JBS 09-APR-1979
56 0056 1 1-007 - Change LIB$S and OTS$S to STR$. JBS 21-MAY-1979
57 0057 1 1-008 - Correct calls to STR$FREE1_DX. JBS 22-MAY-1979

```

 ! End of major frame
 ! File: BASEND.B32 Edit: MDL1018

```

: 58 0058 1 1-009 - Again correct calls to STR$FREE1-DX. JBS 22-MAY-1979
: 59 0059 1 1-010 - Use JSB entry point to STR$FREE1-DX. JBS 11-JUN-1979
: 60 0060 1 1-011 - Improve the error message for the case of an error handler
: 61 0061 1 falling off the end of the program. JBS 24-JUL-1979
: 62 0062 1 1-012 - Call BASS$UNWIND when cutting back GOSUBs. JBS 03-AUG-1979
: 63 0063 1 1-013 - If the frames are confused, give a special message if the
: 64 0064 1 bad frame is a DEF*. JBS 18-SEP-1979
: 65 0065 1 1-014 - Use BASS$Z1 RET to cut back frames. JBS 09-NOV-1979
: 66 0066 1 1-015 - Cut back DEF* frames, too. JBS 09-NOV-1979
: 67 0067 1 1-016 - Add support for run-time dimensioned arrays. PLL 13-May-1982
: 68 0068 1 1-017 - Check frame version number before executing run-time array
: 69 0069 1 code. PLL 17-May-1982
: 70 0070 1 1-018 - recover the pointer to temporary storage after unwinding
: 71 0071 1 GOSUB and DEF* frames. MDL 27-Dec-1983
: 72 0072 1 --
: 73 0073 1
: 74 0074 1
: 75 0075 1 <BLF/PAGE>

```



```
77 0076 1 |
78 0077 1 | SWITCHES:
79 0078 1 |
80 0079 1 |
81 0080 1 | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
82 0081 1 |
83 0082 1 |
84 0083 1 | LINKAGES:
85 0084 1 |
86 0085 1 |
87 0086 1 | LINKAGE
88 0087 1 |     BAS$INIT_LINK = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 2) : !
89 0088 1 |     GLOBAL (BSF$A_MAJOR_STG = 11, BSF$A_MINOR_STG = 10, BSF$A_TEMP_STG = 9) !
90 0089 1 |     NOPRESERVE (8, 7, 6, 5, 4, 3, 2, 1, 0);
91 0090 1 |
92 0091 1 | REQUIRE 'RTLIN:STRLNK'; ! String linkages
93 0276 1 |
94 0277 1 |
95 0278 1 | TABLE OF CONTENTS:
96 0279 1 |
97 0280 1 |
98 0281 1 | FORWARD ROUTINE
99 0282 1 |     BAS$END_R8 : NOVALUE BAS$INIT_LINK; ! end major frame
100 0283 1 |
101 0284 1 |
102 0285 1 | INCLUDE FILES:
103 0286 1 |
104 0287 1 |
105 0288 1 | LIBRARY 'RTLSTARLE'; ! define desc offsets
106 0289 1 |
107 0290 1 | REQUIRE 'RTLIN:RTLPSECT'; ! Macros for defining psects
108 0385 1 |
109 0386 1 | REQUIRE 'RTLIN:BASFRAME'; ! Define frame structure
110 0589 1 |
111 0590 1 | REQUIRE 'RTLIN:BASINARG'; ! Define argument list
112 0674 1 |
113 0675 1 |
114 0676 1 | MACROS:
115 0677 1 |
116 0678 1 |     NONE
117 0679 1 |
118 0680 1 | EQUATED SYMBOLS:
119 0681 1 |
120 0682 1 |     NONE
121 0683 1 |
122 0684 1 | PSECTS:
123 0685 1 |
124 0686 1 | DECLARE_PSECTS (BAS); ! Declare psects for BAS$ facility
125 0687 1 |
126 0688 1 | OWN STORAGE:
127 0689 1 |
128 0690 1 |     NONE
129 0691 1 |
130 0692 1 | EXTERNAL REFERENCES:
131 0693 1 |
132 0694 1 |
133 0695 1 | EXTERNAL
```

```

: 134      0696 1      BAS$$ZI_RET;                ! A RET instruction
: 135      0697 1
: 136      0698 1      EXTERNAL ROUTINE
: 137      0699 1      BAS$$STOP : NOVALUE,          ! signals error
: 138      0700 1      STR$FREE1 DX_R4 : STR$JSB_GETFRE, ! deallocate a string
: 139      0701 1      BAS$$UNWIND : NOVALUE,         ! Unwind a frame
: 140      0702 1      BAS$HANDLER,                  ! flags a BASIC frame
: 141      0703 1      LIB$FREE_VM;                   ! deallocate virtual space
: 142      0704 1
: 143      0705 1      !+
: 144      0706 1      ! The following are the error codes used in this module.
: 145      0707 1      !-
: 146      0708 1
: 147      0709 1      EXTERNAL LITERAL
: 148      0710 1      BAS$K_RETWITGOS : UNSIGNED (8), ! RETURN without GOSUB
: 149      0711 1      BAS$K_ILLEXIDF : UNSIGNED (8), ! Illegal exit from DEF*
: 150      0712 1      BAS$K_ERRTRANEE : UNSIGNED (8), ! ERROR trap needs RESUME
: 151      0713 1      BAS$K_PROLOSSOR : UNSIGNED (8); ! Program lost, sorry
: 152      0714 1

```



```
154 0715 1 GLOBAL ROUTINE BASSEND_R8 (                ! end of major frame
155 0716 1     ARGLIST                                ! original frame parameters
156 0717 1     ) : NOVALUE BASS$INIT_LINK =
157 0718 1
158 0719 1 ++
159 0720 1 FUNCTIONAL DESCRIPTION:
160 0721 1
161 0722 1     Tear down a frame for a BASIC-PLUS-2 major procedure.
162 0723 1     All heap storage is deallocated. The argument is the same
163 0724 1     as for BASS$INIT_R8, for validity checking. Intervening
164 0725 1     GOSUB frames are removed. These frames had better not
165 0726 1     save any registers not saved by the major procedure.
166 0727 1
167 0728 1 FORMAL PARAMETERS:
168 0729 1
169 0730 1     ARGLIST.ra.v    List of information used to set up the
170 0731 1                    frame. See BASIC-PLUS-2/VAX Description
171 0732 1                    of Generated Code for details.
172 0733 1
173 0734 1 IMPLICIT INPUTS:
174 0735 1
175 0736 1     The frame, as set up by BASS$INIT_R8.
176 0737 1
177 0738 1 IMPLICIT OUTPUTS:
178 0739 1
179 0740 1     NONE
180 0741 1
181 0742 1 ROUTINE VALUE:
182 0743 1
183 0744 1     NONE
184 0745 1
185 0746 1 COMPLETION CODES:
186 0747 1
187 0748 1     NONE
188 0749 1
189 0750 1 SIDE EFFECTS:
190 0751 1
191 0752 1     Deallocates the heap storage and virtual arrays local to this
192 0753 1     major procedure.
193 0754 1
194 0755 1 --
195 0756 1
196 0757 2 BEGIN
197 0758 2
198 0759 2 EXTERNAL REGISTER
199 0760 2     BSF$A_MAJOR_STG : REF BLOCK [0, BYTE],
200 0761 2     BSF$A_MINOR_STG : REF VECTOR,
201 0762 2     BSF$A_TEMP_STG : REF VECTOR;
202 0763 2
203 0764 2 BUILTIN
204 0765 2     FP;
205 0766 2
206 0767 2 MAP
207 0768 2     ARGLIST : REF BLOCK [0, BYTE] FIELD (BASS$INIT_ARGS);    ! arg list
208 0769 2
209 0770 2 REGISTER
210 0771 2     FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD),        ! pointer to FCD
```



```
211 0772 2      PREV_FMP : REF BLOCK [0, BYTE] FIELD (BSF$FCD); ! previous FCD
212 0773 2
213 0774 2
214 0775 2      + First cut back any GOSUB frames. We wish to make the presence of
215 0776 2      the GOSUB frame invisible except on traceback.
216 0777 2
217 0778 2      FMP = .FMP;
218 0779 2
219 0780 2      WHILE ((.FMP [BSF$B_PROC_CODE] EQL BSF$K_PROC_GOSB) OR (.FMP [BSF$B_PROC_CODE] EQL BSF$K_PROC_DEFS)) DO
220 0781 2      BEGIN
221 0782 2
222 0783 2      + We have a GOSUB or DEF* frame, remove it.
223 0784 2
224 0785 2      BAS$$UNWIND (.FMP);
225 0786 2      FMP [BSF$A_SAVED_PC] = BAS$$ZI RET;
226 0787 2      PREV_FMP = .FMP [BSF$A_SAVED_FP];
227 0788 2
228 0789 2      IF (.PREV_FMP [BSF$A_HANDLER] NEQA BAS$HANDLER)
229 0790 2      THEN
230 0791 2
231 0792 2      + The previous frame is not a BASIC frame. This is unreasonable
232 0793 2      since DEF*s and GOSUBs should only be callable from inside a BASIC main
233 0794 2      procedure.
234 0795 2
235 0796 2      BAS$$STOP (BAS$K_RETWITGOS);
236 0797 2
237 0798 2      FMP = .PREV_FMP;
238 0799 2      END;
239 0800 2
240 0801 2      + Make sure this is a major frame.
241 0802 2
242 0803 2
243 0804 2
244 0805 2      CASE .FMP [BSF$B_PROC_CODE] FROM BSF$K_PROC_MAIN TO BSF$K_PROC_IOL OF
245 0806 2      SET
246 0807 2
247 0808 2      [BSF$K_PROC_MAIN, BSF$K_PROC_SUB, BSF$K_PROC_EXTF] :
248 0809 2      BEGIN
249 0810 2      +
250 0811 2      + recover the pointer to temporary storage for this frame.
251 0812 2
252 0813 2      BSF$A_TEMP_STG = .FMP [BSF$A_BASE_R9];
253 0814 2      END;
254 0815 2
255 0816 2      [BSF$K_PROC_DEFS] :
256 0817 2      BAS$$STOP (BAS$K_ILLEXIDF);
257 0818 2
258 0819 2      [BSF$K_PROC_DEF, BSF$K_PROC_GOSB, BSF$K_PROC_IOL] :
259 0820 2      BAS$$STOP (BAS$K_PROLOSSOR);
260 0821 2
261 0822 2      [BSF$K_PROC_ONER] :
262 0823 2      BAS$$STOP (BAS$K_ERRTRANEE);
263 0824 2
264 0825 2      [OUTRANGE] :
265 0826 2      BAS$$STOP (BAS$K_PROLOSSOR);
266 0827 2      TES;
267 0828 2
```



```
268 0829 2 +
269 0830 2 | Check to be sure that this is the correct exit. This should
270 0831 2 | never fail, since there can be only one argument list to the
271 0832 2 | major procedure.
272 0833 2 |
273 0834 2 |
274 0835 3 IF (.FMP [BSF$A_INIT_ARG] NEQA .ARGLIST)
275 0836 3 THEN
276 0837 2 +
277 0838 2 | The argument lists are not at the same address. This exit must not
278 0839 2 | correspond to the entry. Signal an error.
279 0840 2 |
280 0841 2 | BAS$$STOP (BAS$K_PROLOSSOR);
281 0842 2 |
282 0843 2 +
283 0844 2 | Deallocate any temporary string storage.
284 0845 2 |
285 0846 2 |
286 0847 2 INCR COUNTER FROM 1 TO .ARGLIST [BAS$L_IN_NO_TST] DO
287 0848 2 STR$FREE1_DX_R4 (BSF$A_TEMP_STG [(COUNTER - 1)*2]);
288 0849 2 |
289 0850 2 +
290 0851 2 | Deallocate local dynamic strings.
291 0852 2 |
292 0853 2 |
293 0854 2 INCR COUNTER FROM 1 TO .ARGLIST [BAS$W_IN_NO_DST] DO
294 0855 2 STR$FREE1_DX_R4 (.FMP [BSF$A_STR_DESC] + ((COUNTER - 1)*(2*%UPVAL)));
295 0856 2 |
296 0857 2 +
297 0858 2 | If there were any run-time dimensioned arrays, deallocate the storage
298 0859 2 | that was allocated for their elements. Notice that the frame became
299 0860 2 | bigger to accomodate run-time arrays - don't try to deallocate anything
300 0861 2 | for previous versions of the frame which did not support run-time arrays.
301 0862 2 |
302 0863 2 |
303 0864 3 IF (.ARGLIST [BAS$B_IN_V_FCD] GTR 1) AND (.FMP [BSF$A_RTA_DESC] NEQ 0)
304 0865 3 THEN
305 0866 3 BEGIN
306 0867 3 LOCAL
307 0868 3 END_ADDR,
308 0869 3 ARRAY_DSC_SIZE,
309 0870 3 RTA_DSC : REF BLOCK [16, BYTE];
310 0871 3
311 0872 3 RTA_DSC = .FMP [BSF$A_RTA_DESC]; ! point at 1st rt array dsc
312 0873 3 END_ADDR = .RTA_DSC + .ARGLIST [BAS$L_IN_LEN_RTA_DT];
313 0874 3 WHILE .RTA_DSC [SS .END_ADDR] DO ! loop thru all rt array dsc's
314 0875 3 BEGIN
315 0876 3 IF .RTA_DSC [DSC$A_POINTER] NEQ 0 ! make sure REMAP allocated space
316 0877 3 THEN
317 0878 3 LIB$FREE VM (RTA_DSC [DSC$L_ARSIZE], RTA_DSC [DSC$A_POINTER]);
318 0879 3 ARRAY_DSC_SIZE = (4 * ! block 1 constant size
319 0880 3 (1 + .RTA_DSC [DSC$B_DIMCT]) + ! block 2, multipliers
320 0881 3 (2 * .RTA_DSC [DSC$B_DIMCT])) * ! block 3, bounds
321 0882 3 %UPVAL; ! compute in bytes
322 0883 3 RTA_DSC = .RTA_DSC + .ARRAY_DSC_SIZE; ! advance to next array dsc
323 0884 3 END;
324 0885 2 END;
```


: 325
: 326
: 327
: 328
: 329
: 330
: 3310886 2
0887 2
0888 2
0889 2
0890 2
0891 2
0892 1+ All done. The 'RET' instruction done by the compiled code
- will cut back the stack, so we don't need to do it here.RETURN;
END;

! of BASSEND_R8

.TITLE BASSEND
.IDENT \1-018\.EXTRN BAS\$\$ZI_RET, BAS\$\$STOP
.EXTRN STR\$FREE1_DX_R4
.EXTRN BAS\$\$UNWIND, BAS\$HANDLER
.EXTRN LIB\$FREE_VM, BAS\$K_RETWITGOS
.EXTRN BAS\$K_ILLEXIDEF
.EXTRN BAS\$K_ERRTRANEE
.EXTRN BAS\$K_PROLOSSOR

.PSECT _BAS\$CODE, NOWRT, SHR, PIC, 2

57		50	DO	00000	BASSEND_R8::				
55		5D	DO	00003	MOVL	R0, R7			0715
06	E5	A5	91	00006	1\$:	MOVL	FP, FMP		0778
		06	13	0000A	CMPB	-27(FMP), #6			0780
05	E5	A5	91	0000C	BEQL	2\$			
		31	12	00010	CMPB	-27(FMP), #5			
		55	DD	00012	BNEQ	4\$			
00000000G	00	01	FB	00014	2\$:	PUSHL	FMP		0785
10	A5	00000000G	00	9E	0001B	CALLS	#1, BAS\$\$UNWIND		
	52	0C	A5	D0	00023	MOVAB	BAS\$\$ZI_RET, 16(FMP)		0786
	50	00000000G	00	9E	00027	MOVL	12(FMP), PREV_FMP		0787
	50		62	D1	0002E	MOVAB	BAS\$HANDLER, R0		0789
			0B	13	00031	CML	(PREV_FMP), R0		
	7E	00G	8F	9A	00033	BEQL	3\$		
00000000G	00		01	FB	00037	MOVZBL	#BAS\$K_RETWITGOS, -(SP)		0796
	55		52	D0	0003E	3\$:	CALLS	#1, BAS\$\$STOP	
			C3	11	00041	MOVL	PREV_FMP, FMP		0798
	01	E5	A5	8F	00043	4\$:	BRB	1\$	0780
001E	0012	0012	0012	00048	5\$:	CASEB	-27(FMP), #1, #7		0805
001E	0024	001E	0018	00050	.WORD	6\$-5\$,-			
						6\$-5\$,-			
						6\$-5\$,-			
						8\$-5\$,-			
						7\$-5\$,-			
						8\$-5\$,-			
						9\$-5\$,-			
						8\$-5\$			
			0C	11	00058	BRB	8\$		0826
59	EC	A5	D0	0005A	6\$:	MOVL	-20(FMP), BSF\$A_TEMP_STG		0813
		17	11	0005E	BRB	11\$			0805
7E	00G	8F	9A	00060	7\$:	MOVZBL	#BAS\$K_ILLEXIDEF, -(SP)		0817
		0A	11	00064	BRB	10\$			
7E	00G	8F	9A	00066	8\$:	MOVZBL	#BAS\$K_PROLOSSOR, -(SP)		0820
		04	11	0006A	BRB	10\$			
7E	00G	8F	9A	0006C	9\$:	MOVZBL	#BAS\$K_ERRTRANEE, -(SP)		0823

00000000G	00	01	FB	00070	10\$:	CALLS	#1, BAS\$\$STOP	
	57	D8	A5	D1	00077	11\$:	CMPL	-40(FMP), ARGLIST
			0B	13	0007B		BEQL	12\$
	7E	00G	8F	9A	0007D		MOVZBL	#BAS\$K_PROLOSSOR, -(SP)
00000000G	00		01	FB	00081		CALLS	#1, BAS\$\$STOP
			56	D4	00088	12\$:	CLRL	COUNTER
			0F	11	0008A		BRB	14\$
50	56		01	78	0008C	13\$:	ASHL	#1, COUNTER, R0
	50	F8 A940	DE	00090		MOVAL	-8(BSF\$A_TEMP_STG)[R0], R0	
		00000000G	00	16	00095		JSB	STR\$FREE1 DX_R4
EC	56	30	A7	F3	0009B	14\$:	AOBLEQ	48(ARGLIST), COUNTER, 13\$
	58	28	A7	3C	000A0		MOVZWL	40(ARGLIST), R8
			56	D4	000A4		CLRL	COUNTER
			0E	11	000A6		BRB	16\$
	50	E0 B546	7E	000A8	15\$:	MOVAQ	0-32(FMP)[COUNTER], R0	
	50		08	C2	000AD		SUBL2	#8, R0
		00000000G	00	16	000B0		JSB	STR\$FREE1 DX_R4
EE	56		58	F3	000B6	16\$:	AOBLEQ	R8, COUNTER, -15\$
	01	04	A7	91	000BA		CMPL	4(ARGLIST), #1
			39	1B	000BE		BLEQU	19\$
		B8	A5	D5	000C0		TSTL	-72(FMP)
			34	13	000C3		BEQL	19\$
	53	B8	A5	D0	000C5		MOVL	-72(FMP), RTA_DSC
54	53	40	A7	C1	000C9		ADDL3	64(ARGLIST), RTA_DSC, END_ADDR
	54		53	D1	000CE	17\$:	CMPL	RTA_DSC, END_ADDR
			26	18	000D1		BGEQ	19\$
		04	A3	D5	000D3		TSTL	4(RTA_DSC)
			0D	13	000D6		BEQL	18\$
		04	A3	9F	000D8		PUSHAB	4(RTA_DSC)
		0C	A3	9F	000DB		PUSHAB	12(RTA_DSC)
00000000G	00		02	FB	000DE		CALLS	#2, LIB\$FREE_VM
	50	0B	A3	9A	000E5	18\$:	MOVZBL	11(RTA_DSC), R0
	50	6040	3E	000E9		MOVAV	(R0)[R0], R0	
52	50		02	78	000ED		ASHL	#2, R0, ARRAY_DSC_SIZE
	52		14	C0	000F1		ADDL2	#20, ARRAY_DSC_SIZE
	53		52	C0	000F4		ADDL2	ARRAY_DSC_SIZE, RTA_DSC
			D5	11	000F7		BRB	17\$
			05	000F9	19\$:	RSB		

; Routine Size: 250 bytes, Routine Base: _BAS\$CODE + 0000

:	332	0893	1	
:	333	0894	1	END
:	334	0895	1	
:	335	0896	0	ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
_BAS\$CODE	250	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	3	0	581	00:01.1

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:BASEND/OBJ=OBJ\$:BASEND MSRC\$:BASEND/UPDATE=(ENH\$:BASEND)

: Size: 250 code + 0 data bytes
: Run Time: 00:09.3
: Elapsed Time: 00:19.5
: Lines/CPU Min: 5755
: Lexemes/CPU-Min: 22561
: Memory Used: 111 pages
: Compilation Complete

0022 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASEDDFS
LIS

BASERROR
LIS

BASEDDDF
LIS

BASEDIT
LIS

BASEND
LIS

BASEDUP
LIS

BASEMULP
LIS

BASEDDGB
LIS

BASERTXT
LIS